Rwanda Journal of Engineering, Science, Technology and Environment, Volume 6, Issue 1, June 2024/eISSN: 2617-233X | print ISSN: 2617-2321

Segmentation of Prostate Cancer in MRI Using Deep Learning

Muhayimana Odette[1]

[1] University of Rwanda/College of Science and Technology, Postal address: 3900 Kigali, Rwanda
*Corresponding Author: muhayiodette06@gmail.com

Abstract

Prostate cancer is one of the most causes of cancer deaths for men worldwide and it is still a significant public health problem. However, prostate cancer can be cured if detected at an early stage. Due to its ability to produce detailed anatomical structures, Magnetic Resonance Imaging (MRI) is one of the most used modalities for prostate cancer diagnosis and treatment. The accurate segmentation of the prostate from MRI is crucial for diagnosis and treatment planning of prostate cancer. Deep learning has provided important support in early disease detection, and image processing analysis, especially in image classification, image registration, image segmentation and medical treatment plans. In this paper, we propose an automatic segmentation of the prostate in MRI based on deep learning methods. A convolutional Neural Network special type named 3D U-Net is used to segment the prostate in MRI. We conducted the 10-fold cross-validation experiments on the public Promise 12 Data set of 50 prostate images and achieved a mean Dice similarity coefficient of 84.92% and a mean Hausdorff distance of 5.3 mm. The experiments proved that the proposed algorithm performed with promising results.

Keywords: Convolutional neural network, deep learning, magnetic resonance imaging, prostate segmentation.

1. Introduction

Prostate cancer is one of the top diseases causing deaths among men, and it was ranked second to lung cancer as the most frequent cancer death leading disease in men worldwide (Adeloye et al., 2016). This cancer results from the uncontrolled multiplication of prostate healthy cells, forming a tumor that is identified as benign or malignant. Some types of prostate cancer grow slowly and do not spread out of the prostate gland while the other category behaves aggressively and metastasize to other parts surrounding the prostate gland. However, the complicated anatomical structure due to its unclear boundary at the apex and the base; and its varying shape make its surgery operation a challenge when it is the only treatment to assign to the patient.

In 2018, prostate cancer in men counted 1, 276, 106 new cases and 358, 989 deaths i.e. 3:8% of all deaths caused by cancer in men all over the world. The GLOBCAN 2020 reported an estimate of 1, 414, 259 or 14:1% of new cases with 375, 000 deaths of prostate cancer worldwide, which ranks this disease as the second most frequent cancer and the fifth leading cause of cancer death among men in 2020. The late detection of prostate cancer is at the base of the observed high

https://dx.doi.org/10.4314/rjeste.v6i1.4

mortality rate. As published by cancer.net, studies have shown that 87% of men treated when their cancer is diagnosed early, can have a life expectancy of five years.

The volume of the prostate is the key indicator of the prostate's health because it reveals crucial information about the stage of the prostate cancer, and the probable prognosis and gives a direction to urologists about possible treatments to assign to the prostate cancer holder. Moreover, the volume of the prostate provides experts with useful information to avoid over-diagnosis and over-treatment of a slow tumor which could be dangerous to the man's life. The segmentation operation on the T2-weighted (T2W) prostate magnetic resonance images (MRIs) achieves good results for accurate treatment planning and provides higher support to the automated prostate cancer diagnosis algorithms. It also provides crucial guidance in various existing and developing clinical applications. For instance, radiotherapy planning for prostate cancer treatment relies on the proper delineation of the prostate in imaging modality which is mostly MRI.

A range of methods have been proposed to segment the prostate gland by using feature-based machine learning. For instance, (Maan et al., 2012) proposed a segmentation approach using multi-spectral MRI such as T1, T2 and proton density-weighted images and used the non-parametric and parametric classifiers, Bayesian-quadratic, and the K-nearest neighbor for prostate segmentation. (Khan et al., 2020) worked on the evaluation of four encoder-decoder CNNs in the segmentation of the prostate gland in T2W magnetic resonance imaging (MRI) images. (Rundo et al., 2019) carried out a study to evaluate the generalization ability of CCN-based architectures on three T2-Weighted MR datasets, each one consisting of a different number of patients and heterogeneous image characteristics, collected by different institutions. (Zhang et al., 2017) proposed an automatic pipeline to segment the prostate gland in diffusion magnetic resonance images (dMRI), where he applied a clustering method to the dMRI to separate the prostate gland from the surrounding tissues. (Habes et al., 2013) proposed an automatic prostate segmentation in whole-body MRI scans for epidemiological studies. This method is based on the SVM to detect the prostate on MRIs and uses SVM binary classifier to segment the 3D MRI voxels. Then the resulting automatic 3D features such as median, gradient, anisotropy and Eigenvalues of the 3D tensor were used to create the classification binary mask for the notation process. Later, deep learning methods emerged to solve more complex and huge data. Wang proposed a classic Generative Adversarial Network-based automatic segmentation method named SegGAN to segment the prostate from MRI. (Tian et al., 2021) proposed an end-to-end prostate CNN-based segmentation method he named PSNet to segment the prostate from MRI; and many more.

The main purpose of this study is to show the effectiveness of the proposed automatic segmentation of the prostate in MRI compared with manual segmentation and other automatic segmentation methods. We also aim to create a U-Net architecture that is capable of extracting features from the input image that help to capture its context and then provide a segmented image which is the same as the input image but with a correct localization of the information that can guide medical doctors monitoring the prostate cancer prognosis and staging, treatment plan and surgery operation. The advantage of this proposed segmentation method is that it can avoid subjective judgments and time-consuming processes.

2. Materials and Methods

This chapter introduces in detail the methodologies including MRI, CNN and U-Net that are involved in implementing the target prostate segmentation model from MRI as designed in the figure figure3. The goal is to concentrate and keep it as intuitive and simple as possible on the principles of magnetic resonance imaging.

https://dx.doi.org/10.4314/rjeste.v6i1.4

2.1. Magnetic Resonance Imaging (MRI)

Medical imaging is defined as a collection of methods and mechanisms for generating visual representations of the body's inner organs for clinical analysis and medical intervention as well as the visual representation of the role of certain tissues and organs. Medical imaging attempts to expose internal structures hidden by the skin and bones as well as for disease diagnosis and treatment planning. Medical imaging also creates a normal anatomy and psychology database to facilitate the identification of abnormalities. Computer Tomography (CT), Magnetic Resonance Imaging (MRI) and Positron Emission Tomography (PET) are the key modalities in medical imaging (Rundo et al., 2019).

2.2. Convolution Neural Network (CNN)

Deep learning has attracted many researchers' interests in recent years and CNN has proved the most outperforming algorithm among different deep learning models (Yamashita et al., 2018). CNN is a class of artificial neural networks that has attracted many researchers' attention in computer vision tasks due to the highly promising results shared in the object recognition competition named ImageNet Large Scale Visual Recognition Competition (ILSVRC) in 2012 (Krizhevsky et al., 2012), (Russakovsk et al., 2015). In addition, CNN has achieved great results in various medical research fields. For instance, (Gulshan et al., 2016) and (Esteva et al., 2017) proved the potential of deep learning for retinopathy screening, skin lesion classification and lymph node metastasis detection respectively. Moreover, many publications have been made in several areas among others being lesion detection (Lahkan et al. 2017), classification (Yasaka et al., 2018), segmentation [4], image reconstruction (Kim et al. 2018), (Liu et al. 2018), and natural language processing (Chen et al., 2018). CNN is designed to automatically adapt to learning spatial hierarchies of features from low to high-level patterns. Mathematically, CNN is defined as an architecture built upon three main building blocks/layers: convolution, pooling and fully connected layers. The convolution and pooling layers play an important role in extracting features from input data while the fully connected layer maps the extracted features into the final output. At the convolution layer, a bunch of operations are performed on the 2D or 3D images depending on whether the network is a 2D- or 3D-CNN. These operations are followed by a small grid/an array of parameters named kernel that acts as an optimizing feature extractor from the image at each position. These two operations, i.e. convolution and kernel make the CNN an efficient method for image processing because a feature can occur anywhere in the image. Along with feeding each layer's output as input to its next layer, the number of extracted features grows progressively and makes the learning process complex. The parameter optimization process is performed to minimize the difference between outputs and ground truth through back-propagation and gradient descent.

2.2.1. Reasons for choosing CNN

The key difference between CNN and traditional radio-mics studies resides in that the latter mostly use hand-crafted feature extractor methods including texture analysis, followed by traditional machine learning classifiers such as Random Forest and Support Vector Machine (Razzak et al., 2018), (Christ et al., 2016) whereas CNN does not require any hand-crafted feature extraction. Moreover, CNN does not require human expertise to segment an organ or tumor and it is computationally expensive because it learns from huge amounts of data. Therefore, CNN models require Graphical Processing Units (GPUs) to increase the model training speed.

https://dx.doi.org/10.4314/rjeste.v6i1.4

2.2.2. CNN Architecture

The CNN architecture is made up of repetitive stacks of different convolution layers and a pooling layer, followed by one or more fully connected layers. The step at which the input data is transformed into output is called forward propagation. According to the loss between the output and label, the backpropagation is used to learn the convolution kernel coefficients and weights.

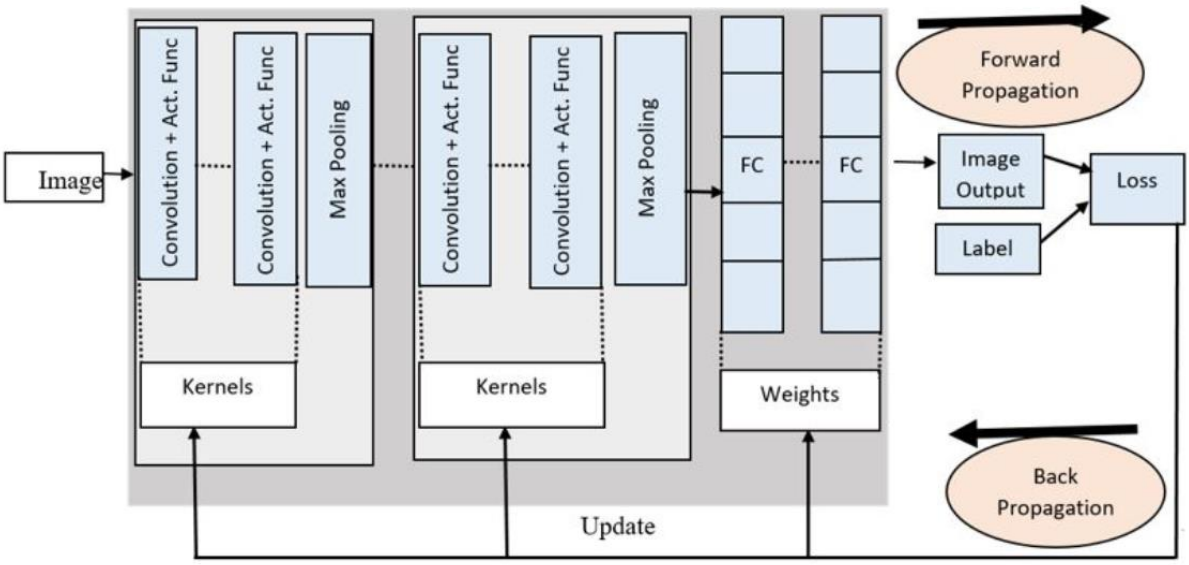

Figure 1. The basic architecture of CNN

❖ **Convolution layer**

The convolution layer is the basic building block of CNN. Its main function is to perform feature extraction on the input image through the combination of linear and non-linear operations: convolution operation and activation function. The idea of stacking a bunch of convolution layers in a CNN model allows the layers close to the input to learn low-level features (e.g., lines) and the layers deeper in the network to learn abstract/high-level features such as shapes.

❖ **Convolution operation**

The convolution operation applies the learned filters known as kernels to extract features from the input image, to create feature maps that richly summarize the features present in the input. At each tensor location, the element-wise product between each kernel element and the input tensor is determined and summed to obtain the output value at the corresponding output tensor location referred to as the function map. This process is performed repeatedly by applying multiple kernels to form an arbitrary number of feature maps representing different input tensor characteristics. Different kernels may therefore be regarded as different extractors of features. Note that the size and number of the kernel are two primary hyper-parameters that describe the convolution process. Frequently, the former is 3x3 and the latter is random, defining the depth of maps of output features (Yamashita et al.2018).

Weight sharing is the main feature of a convolution operation by which kernels are shared across all positions in the image and generate the characteristics among others being: 1) making local feature patterns extracted by translation of kernels invariant as kernels move through all image positions and recognize local learned patterns. 2) by down-sampling in combination with a

https://dx.doi.org/10.4314/rjeste.v6i1.4

pooling operation, learning spatial hierarchies of feature patterns, resulting in an increasingly wider field of view, and 3) increasing model performance by reducing the number of parameters to be learned compared to fully connected networks.

Briefly, the training of a CNN model concerning the convolution layer aims at finding the kernels that perform optimally based on a given training data set for a given task. The kernels are the only parameters that are automatically learned in the convolution layer during the training process, on the contrary, the size of the kernels, the number of kernels, padding and stride are the hyper-parameters that are to be set before starting the training process.

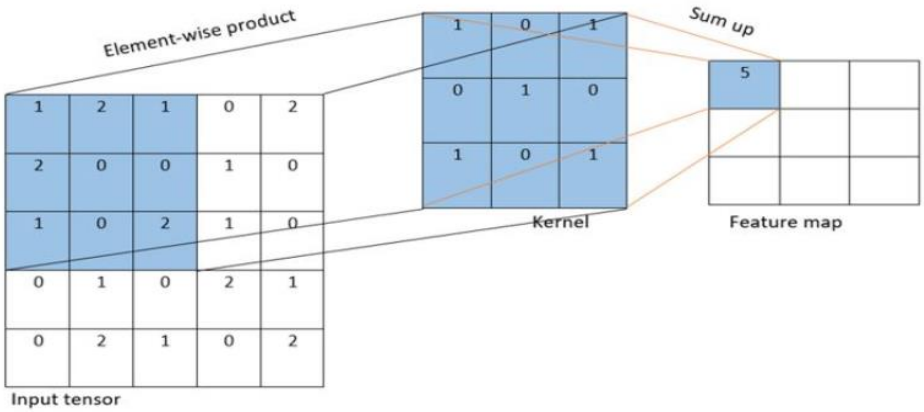

Figure 2: Example of Convolution-step1

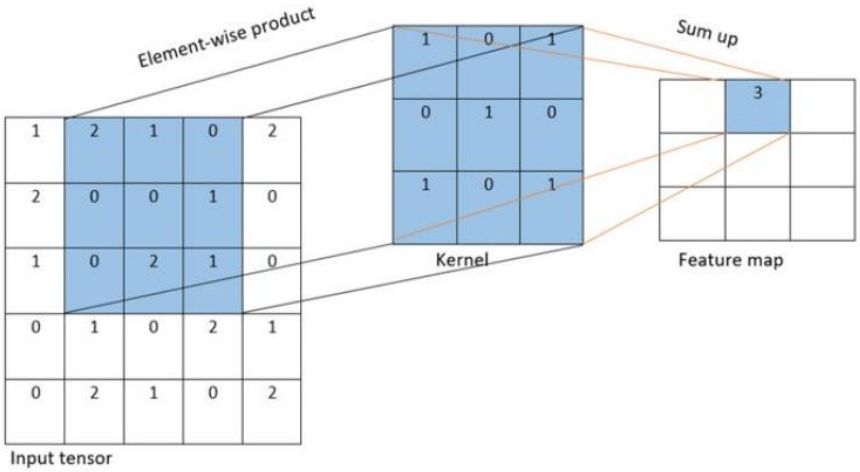

Figure 3: Example of Convolution-step2

https://dx.doi.org/10.4314/rjeste.v6i1.4

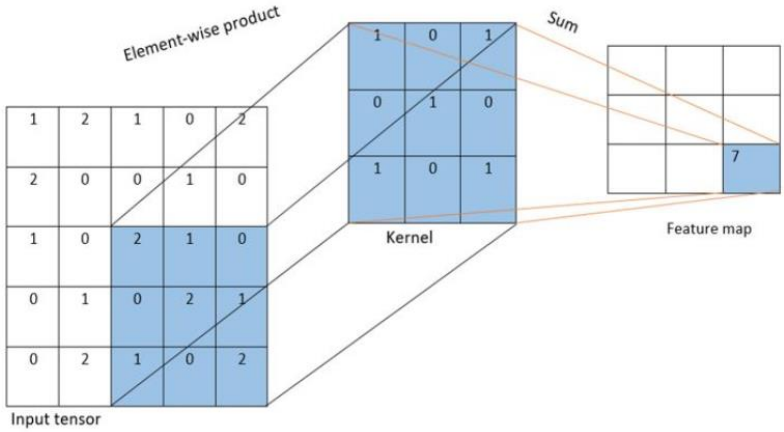

Figure 4: Example of Convolution-step3

2.3. Activation functions

The activation function is a non-linear function that operates on the outputs it receives from linear operations such as convolution. Various non-linear functions including sigmoid or hyperbolic tangent function have been used in the past decades, but the Rectified Linear Function (ReLU) has a lot of popularity over other activation functions in deep learning since it does not activate all neurons at the same time, which speeds up the training process.

2.3.1. Sigmoid function

The sigmoid function often referred to as the logistic function, is a non-linear activation function which is preferably used in the feed-forward neural networks. However, the sigmoid function has the downside of producing a slow model convergence, causing sharp damp gradients during the propagation. The sigmoid function computes the function given by the equation (1).

$$f(x) = \frac{1}{1+e^{-x}} \qquad (1)$$

2.3.2. Softmax function

The softmax function is a type of activation function also used in neural computing, especially in multi-class models where it consists of producing the output whose probability ranges between 0 and 1 with a total sum of the probabilities being equal to 1.

The softmax is given by the relationship given by the equation (2):

$$f(x) = \frac{\exp(x_i)}{\sum_j \exp(y_j)} \qquad (2)$$

Among the probabilities returned by softmax, the target class is displayed with the highest probability. The softmax function is mostly applied to the output layers and differs from the sigmoid function in that it is applied in multivariate classification problems while the sigmoid function is applied in binary classification tasks.

https://dx.doi.org/10.4314/rjeste.v6i1.4

2.3.3. Rectified Linear Unit (ReLU)

Proposed by Nair and Hintson 2010, ReLU is the most widely used activation function in deep learning areas where it has proved to achieve great accuracy. ReLU has many advantages over its counterparts, including being the fastest learning activation function and offering the best performance and model generalization ability. ReLU has its unique property of representing a nearly linear function, which allows it to preserve the properties of linear models; this made them easy to optimize using gradient descent algorithms. In addition, ReLU can alleviate the vanishing gradient problem observed in other activation functions mentioned above (Yamashita et al.2018).

The ReLU activation function performs a threshold operation on each input element where values less than zero are set to zero. ReLU computes the function in equation (3):

$$f(x) = \max(0, x) = \begin{cases} x_i, & if\ X_i \geq 0 \\ 0, if\ X_i < 0 \end{cases} \qquad (3)$$

This means that the neurons will only be deactivated if the output of the linear transformation is less than 0 since this function counts only values that are greater than or equal to zero.

- **Pooling layer**

The pooling layer is a new layer that comes just after the non-linear function like ReLU to reorder the layers present in the CNN that may be repeated one or many times in a model. The pooling layer is added in the network to operate upon every feature map separately to create a new set of the same number of pooled feature maps. The pooling operation is chosen much like a kernel to operate on the feature maps to reduce its size. The most used pooling operation is of kernel size 2x2 applied with a stride of 2 pixels. It is of note that contrary to the filter/kernel, the pooling operation is specified rather than learned. Average and maximum pooling are the common functions used in the pooling operation. Let the below matrix in Table 1 be one of the 4x4 pixels feature maps from the convolution layer.

Table 1: A 4x4 feature map

20	10	20	182
1	95	78	40
10	14	10	4
12	12	40	6

- **Maximum pooling**

Maximum pooling is shortened as max pooling extracts patches from the input feature maps by getting the maximum value in each patch and ignoring all other values, down-sampling the height and width but keeping the depth dimension of the feature map unchanged.

https://dx.doi.org/10.4314/rjeste.v6i1.4

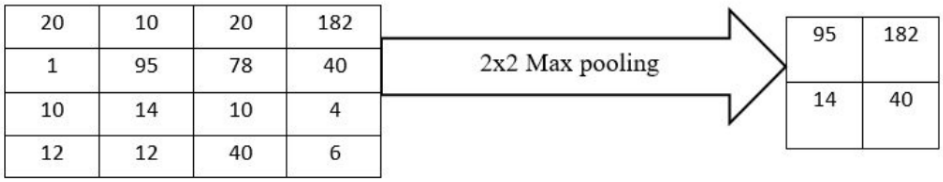

Figure 5: Max pooling

- **Average pooling**

Also referred to as global average pooling, the average pooling operates on the input feature maps to extract patches from the extreme feature maps whereby, the feature map is down-sampled into a 1x1 array of numbers, considering the average of all elements in each feature map, and leaving the depth of feature maps unaltered. The average pooling is preferably used towards the end of the network, just before the fully connected layers where it offers a great advantage to the network of decreasing the number of learnable parameters and allowing the CNN to receive inputs with variable sizes.

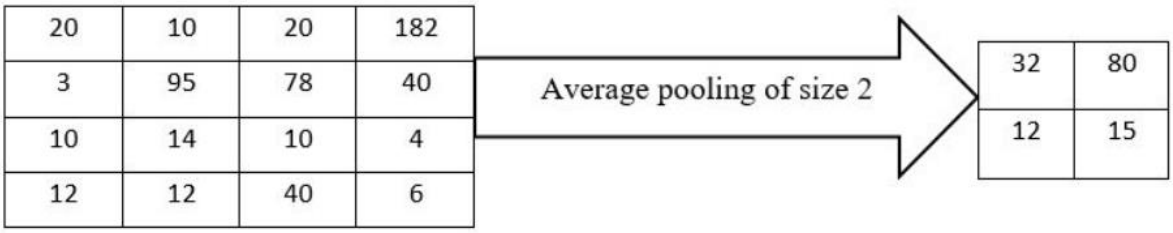

Figure 6: Average pooling

The down-sampled or pooled feature maps resulting from the pooling operation make a summarized version of the features detected in the input and are of paramount importance since small changes in the location of the feature in the input detected by the convolution layer will give a pooled feature map with the feature in the same location thanks to the invariance translation introduced by the pooling operation. i.e., the input translation by a small change does not change the values of the pooled outputs.

2.4. Loss Function

During the model training process, the optimizer, i.e. Adam optimizer in our case, repeatedly tries a set of parameters known as weights and biases until it finds the optimal values for the model to produce the accurate output image. The loss function comes in to help the optimizer assess whether it is trying good values and if the model is achieving good accuracy with promising results as the training goes on.

In the case of our proposed algorithm, we used the Dice Loss function, a region-based loss function introduced in computer vision by (Milletari et al. 2016) for 3D medical image segmentation to determine the similarity between the manually segmented image, i.e. the ground truth and the automatic segmentation image output. This Dice Loss function is inspired by the Dice Coefficient, which is a measure of similarity between two samples and whose value ranges from 0 to 1. The greater the value the better the similarity. when the two samples are perfectly similar, the

https://dx.doi.org/10.4314/rjeste.v6i1.4

dice coefficient takes its value to 1, otherwise, it takes it to 0, meaning that the two samples are completely different. Thus, the Dice Loss function is calculated by the formula: 1-Dice coefficient or it to maximize the similarity between the samples.

The Dice Coefficient (DSC) is defined as shown below:

$$DSC = \frac{2\sum_i^N pigi}{\sum_i^N P_i{}^2 + \sum_i^N g_i{}^2} \quad (4)$$

The **Dice Loss Function** is in turn given by the formula in the equation below:

$$D_{Loss} = 1 - \frac{2\sum_i^N pigi}{\sum_i^N p^2{}_i + \sum_i^N g^2{}_i} \quad (5)$$

where: p_i & g_i represent the pairs of pixel values of prediction and ground truth, respectively.

Note that the Dice Loss function has proven to outperform other loss functions in medical image segmentation because it has its uniqueness of considering the loss information both locally and globally which is very important for a model to achieve a high accuracy.

2.5. U-Net

The developed U-Net architecture comprises two main parts for it to successfully segment the prostate from MRI. Those parts are the contracting path also referred to as the encoder path and the expansive path referred to as the decoder path. To train and evaluate the performance of the model itself, the 10-fold cross-validation approach is used then compute the mean dice score over all the model dice scores of the 10-fold cross-validation. The proposed method is also compared to other segmentation methods presented in the literature by comparing their qualitative and quantitative metrics. Given that the prostate volume varies from patient to patient and that the 3D network requires a common volume size, there should be a selection of a fixed number of total slices around the predicted volume. The total number of slices used in our model is 64. U-Net has an elegant architecture built upon the contracting path and its symmetric path known as the expansive path.

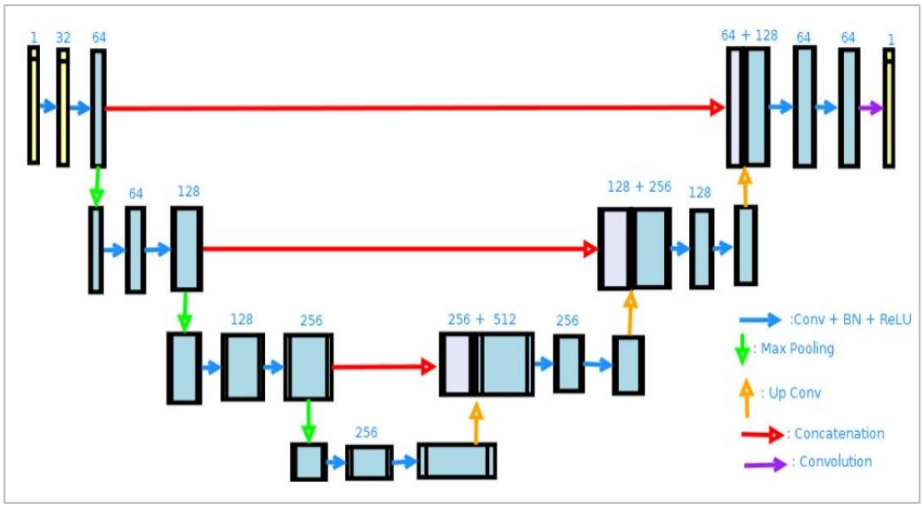

Figure 7: The proposed 3D U-Net model architectural design

https://dx.doi.org/10.4314/rjeste.v6i1.4

The architecture comprises two paths:

The left path commonly called the contracting path or encoder whose task is to down-sample the input image by extracting features from it and reducing its resolution using an appropriate stride. The contracting path follows the standard CNN architecture. It consists of repeated application of Convolution, Batch normalization with Activation function, followed by a 2x2x2 max pooling operation with a stride of 2 for down-sapling. It is key to note that at every stage in the contacting path, the number of feature channels is doubled while the spatial dimensions, height and width are halved.

The right path known as an expansive path or decoder aims at up-sampling the feature map and halving the number of feature channels until the output image has the same size as the input image. The first operation performed on the expansion path is up-sampling, which makes the size of the feature map increase gradually until it reaches the original size of the input. The 2x2x2 up-convolution for up-sampling is followed by two 3x3x3 convolution operations.

After every convolution operation, border pixel information is lost and this can affect the segmentation accuracy. Therefore, after each 3x3x3 up-convolution a concatenation operation is performed to forward the features extracted from the early stages of the contracting path to the expansive path for better information localization. Then after localizing the information, the segmented image is displayed through the last 1x1x1 Convolution layer.

Batch normalization is performed during the training process to avoid the bottleneck and facilitate fast network convergence. Batch normalization is a method of normalizing the data directly during the neural network training after the weighted sum operation and before passing data through the activation function. In this study, each batch is normalized during the training process for all data to be on the same scale.

1) Layers architecture at the down-sampling path:

Convolution layer #1 applies 32, 3x3x3 kernel filter, Batch Normalization followed by ReLU Activation Function

Convolution layer #2 applies 32, 3x3x3 kernel filter, Batch Normalization followed by ReLU Activation Function

Pooling #1 applies a max pooling with a 2x2x2 kernel filter

Convolution layer #3 applies 64, 3x3x3 kernel filter, Batch Normalization followed by ReLU activation function

Convolution layer #4 applies 64, 3x3x3 kernel filter, Batch Normalization followed by ReLU activation function

Pooling layer #2 applies a max pooling with 2x2x2 kernel filter

https://dx.doi.org/10.4314/rjeste.v6i1.4

Convolution layer #5 applies 128, 3x3x3 kernel filter, Batch Normalization followed by ReLU activation function

Pooling layer #3 applies a max pooling with 2x2x2 kernel filter

2) Layers architecture at the bridge:

Convolution layer #7 applies 256, 3x3x3 kernel filter, Batch Normalization followed by ReLU activation function

Convolution layer #8 applies 256, 3x3x3 kernel filter, Batch Normalization followed by ReLU activation function

3) Layers architecture at the up-sampling path:

Transposed convolution layer #1 applies 512, 3x3x3 kernel filter, Batch Normalization followed by ReLU Up-convolution layer #1 applies 768, 3x3x3 kernel filter, Batch Normalization on 256, followed by ReLU activation function

Up-convolution layer #2 applies 256, 3x3x3 kernel filter, Batch Normalization followed by ReLU activation function

Transposed convolution layer #2 applies 256, 3x3x3 kernel filter, Batch Normalization, followed by ReLU activation function

Up-convolution layer #3 applies 384, 3x3x3 kernel filter, Batch Normalization on 128, followed by ReLU activation function

Transposed convolution layer #3 applies 128, 3x3x3 kernel filter, Batch Normalization, followed by ReLU activation function

Up-convolution layer #5 applies 192, 3x3x3 kernel filter, Batch Normalization on 64, followed by ReLU activation function

Transposed convolution layer #4 applies 64, 3x3x3 kernel filter, Batch Normalization, followed by ReLU activation function

Up-convolution layer #6 applies 64, 3x3x3 kernel filter, Batch Normalization, followed by ReLU activation function

The last convolution layer applies 64, 1x1x1 kernel filter with a sigmoid function, then gives the segmented image output.

https://dx.doi.org/10.4314/rjeste.v6i1.4

3. Experiments

3.1. Database description

The data set used, Promise12 consists of 50 T2-Weighted Three-dimensional Magnetic Resonance images (3D MRIs) made publically available by MICCAI Grand Challenge Team for prostate segmentation purposes. The data set includes both patients with benign diseases (for example: benign prostatic hyperplasia) and prostate cancer. For the generalization ability of the algorithm, the data was collected from multiple centers and multiple MRI device vendors with different acquisition and scanning protocols. Figure 8 illustrates some examples of MR images acquired with various protocols and from different areas for better model generalization.

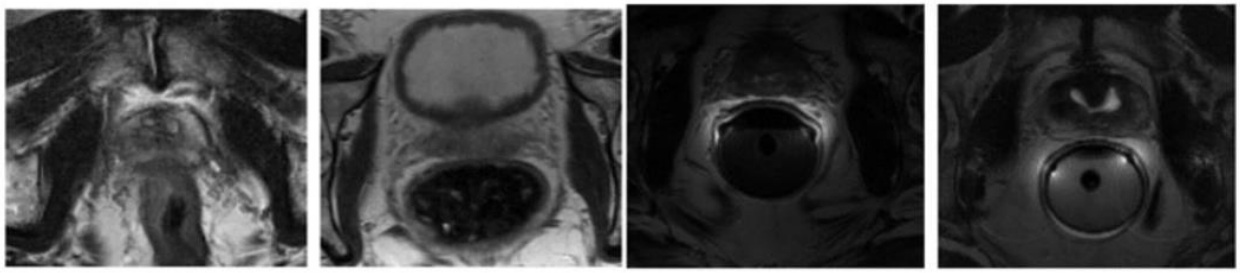

Figure 8: Examples of prostate 3D MR images from Promise12 data set

The data set was split into training and test data sets whereby ninety per cent was used for training the model and the other remaining ten percent was used for testing the model. The ten-fold cross-validation approach was used to train and evaluate the model.

3.2. Data pre-processing

In the pre-processing process of our data, we unify the spatial resolution of each volume to 1.0 x 1.0 x 1.5 millimeter. All volumes processed by the network have a fixed size of 64 x 128 x128. The normalization is carried out on the data to convert all the data to the same scale within a range of [0-1].

3.3. Experimental design

The U-Net architecture proposed in this study is the extension of the U-Net architecture from Ronnerberger et al. and replaces all the 2D operations with their matching 3D counterparts. Our 3D U-Net architecture for prostate segmentation from MR images performs segmentation on the 3D T2-Weighted images using 3D operations including 3D convolutions, 3D max pooling and 3D up-convolution layers. The algorithm implementation was done using Python programming language on the open-source PyTorch framework and the computations were performed with NVIDIA Tesla V100 32G.

A set of parameters and hyper-parameters used for the proposed model is presented and the model performance is evaluated depending on those parameters because the latter play important roles in defining the general learning capacity of a model.

https://dx.doi.org/10.4314/rjeste.v6i1.4

The learning rate expresses the amount that the weights are updated during the training of neural networks for the model for mapping correctly the inputs to outputs in the training data set. It is worth noting that a too-large learning rate allows a model to learn faster but may cause it to diverge and the optimizer can overstep the minima due to high changes in weights. On the other hand, a smaller learning rate allows a model to learn a more optimal set of weights but takes a lot of time to converge due to the steps towards the minimum loss function being tiny. In other words, a smaller learning rate leads to the training model over-fitting, a larger one allows the training model generalization whereas a too-large learning rate causes the model to diverge. The initial learning rate is 0.002 which decreases by a decay of 0.001 after 10, 30, and 60 epochs.

Batch size describes the number of training samples used in one iteration. The batch size choice highly depends on the hardware's memory and on the execution time of the training. The batch size is 2. Weight decay is a concept that helps generalize a model not on the side of data things but on the side of model things. During the model training process in real-world practice, there may be a bunch of data that cannot fit well with a straight line and the idea of increasing the degree of polynomial makes a model more complex at some point and then starts over-fitting. To prevent the model from facing this problem of complexity, the number of parameters has to decrease. However, this does not provide a solution for the real-world application. The introduction of **weight decay** solves the problem by providing the ability to add a lot of parameters to the loss function and at the same time prevent the model from getting complex. Some of the added parameters can be negative and others positive, which can lead to a huge loss of function. The weight decay is multiplied by the sum of squares of the added parameters and this process helps generalize the model even better than using the data augmentation approach.

Mini-batch size expresses the number of samples of the data that are fed to the network from a huge data set. Feeding the whole data set to the network (referred to as batch gradient descent) would be practically inefficient since it is time-consuming and causes the model to get stuck before completing the learning process. Therefore, the mini-batch (mini-batch gradient descent) is practically efficient as it consists of involving various random samples that are greater than 1 and less than the whole data set in the model variable updating process. These samples of data are referred to as batches. The typical batch size is 32 and setting a too big batch size leads to an overgeneralized model, which will not provide good results when new data are presented. Briefly, the batch size means the whole data set is fed into the training process, while the mini-batch size consists of several small training samples that are involved in the training iteration.

Epoch defines how many times the algorithm trains on the data set as a whole. Deciding on the number of epochs depends on the kind of data in hands and the task to accomplish. Different possibilities of choosing the number of epochs for training a model could be either imposing a condition for a model to stop the training process once the error starts tending to zero or to start with a quite small number of epochs and increase it along with the training progress, keeping track of important evaluation metrics such as accuracy. The number of epochs is 100. The below table table2 gives a summary of hyper-parameters and their values defined in our experiment.

https://dx.doi.org/10.4314/rjeste.v6i1.4

Table 2: The model Hyper-parameter values

Hyper-parameter	Value
Optimizer	Adam
Initial Learning	0.002
Weight decay	0.001
Epochs	100
Batch size	2

3.4. Evaluation metrics

The evaluation metrics are an important key to the development of successful algorithms since they provide developers with a way to estimate how well the developed algorithm outputs the best results for a specific task, how it validates its performance on data and how outperforming it is against other approaches. In image segmentation tasks, the popular approach used to evaluate the model is comparing its results with gold standards referred to as manual segmentation results. In the evaluation of the proposed segmentation algorithm, the overlap-based method named Dice's similarity Coefficient (Dice) and a Surface distance-based measure, Hausdorff Distance are used to evaluate the performance.

The qualitative evaluation consists of comparing the gold standard which we consider as the actual image, resulting from the manual segmentation performed by expert physicians with the automatic segmentation results regarded as the prediction obtained from the automatic segmentation by the proposed method. Then comment on how similar they look on the figures.

3.4.1. Dice Similarity Coefficient (DSC)

Dice Coefficient is a statistical tool which measures the similarity between two finite sets of data. It has become conceivably the most broadly used tool in the evaluation of image segmentation algorithms. The dice coefficient is a scalar or numeric vector that ranges between [0, 1], where a similarity of 1 means that the automatic segmentation results and the corresponding gold standard are a perfect match. Mathematically, the Dice Coefficient is computed as:

$$DSC = \frac{2|X \cup Y|}{|X \cap Y|} \quad (6)$$

Where X and Y are the segmented result and gold standard, respectively.

3.4.2. Hausdorff Distance (HD)

In pattern recognition and computer vision problem solving, the challenging task is to determine how much shapes differ from each other. The shape comparison must obey metric properties. The Hausdorff Distance is an evaluation metric that aims at measuring the degree of mismatch between two finite sets by measuring the distance of the point of X that is farthest from any point of Y (Huttenlocher et al., 1993). The Hausdorff Distance is computed by the equation:

https://dx.doi.org/10.4314/rjeste.v6i1.4

$$H(X,Y) = \max(h(X,Y), h(Y,X)) \quad (7),$$

with x and Y representing segmentation results and ground truth respectively. And ||. || are the norm of the sets X and Y. h (X, Y) plays the role of identifying a pixel xX that is far from any pixel of Y and measures the distance from this x to its nearest neighbor in Y using the norm; then ranks each pixel of X based on its distance to the nearest point of Y. It considers the largest ranked pixel as the distance from the most mismatched pixel of X.

3. Results

To start the experiment, the T2-Weighted 3D MR image is fed into the U-Net architecture. At the beginning of the training process, the training loss was high since the model had just learned from very little data. For instance, at the first 10 epochs, the train loss was high while after 30 epochs the learning rate decreased to 0:00000001, see Figure 9 below, which plots the learning rate variation along with the model training process.

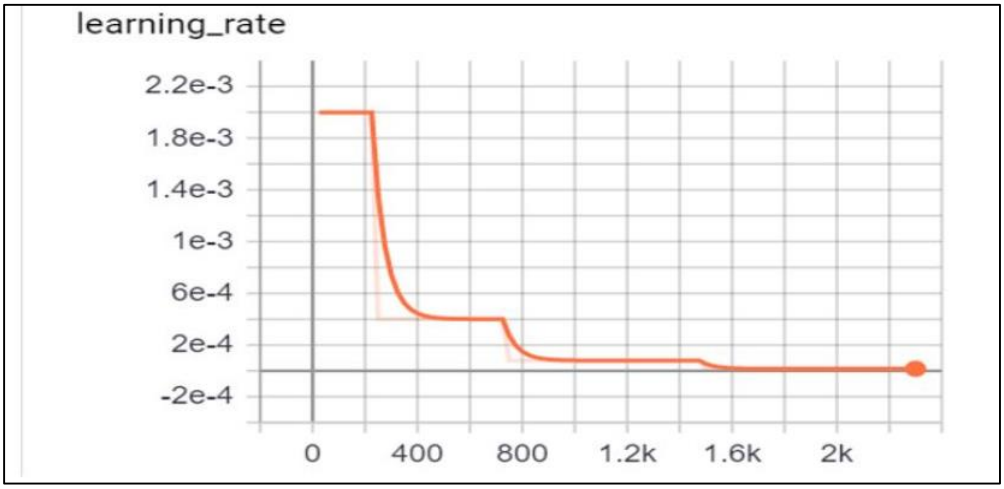

Figure 9: Learning rate variation

The train loss variation in Figure 10 shows how higher the train loss is at the beginning of the training process and the way it decreases along with the training progress. Figure 11 shows the train evaluation score increasing with the learning process.

https://dx.doi.org/10.4314/rjeste.v6i1.4

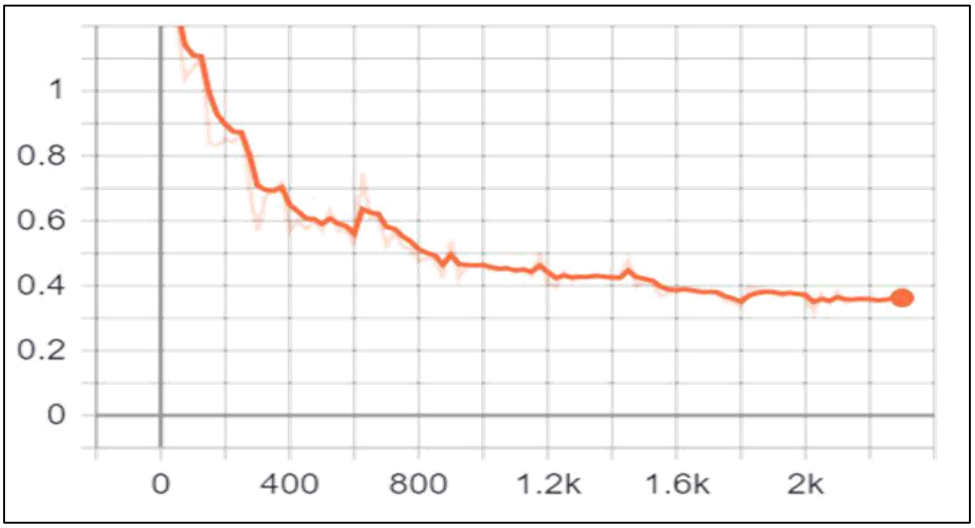

Figure 10: Train Loss variation

The training evaluation score as plotted in Figure 11, is low when the mode starts the training and as the training progresses learning from a lot of data the evaluation score increases and when the satisfying accuracy is reached, the score increases slowly and tends to remain constant.

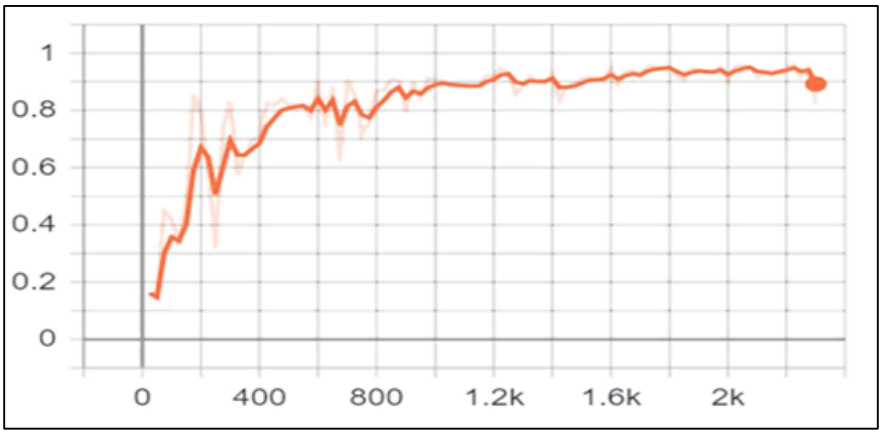

Figure 11: Train Evaluation score

3.1. Quality evaluation results

Figure 12 shows the prostate images manually segmented with their maps automatically segmented. The manual segmentation gold standard is in blue and the automatic segmentation result is in red. For the quality evaluation of the proposed model, we compare the gold standard which we consider as the actual image with the automatic segmentation results regarded as the prediction. The images in Figure 12 show that the automatic segmentation has been performed successfully since both the actual value and predicted value look almost the same.

https://dx.doi.org/10.4314/rjeste.v6i1.4

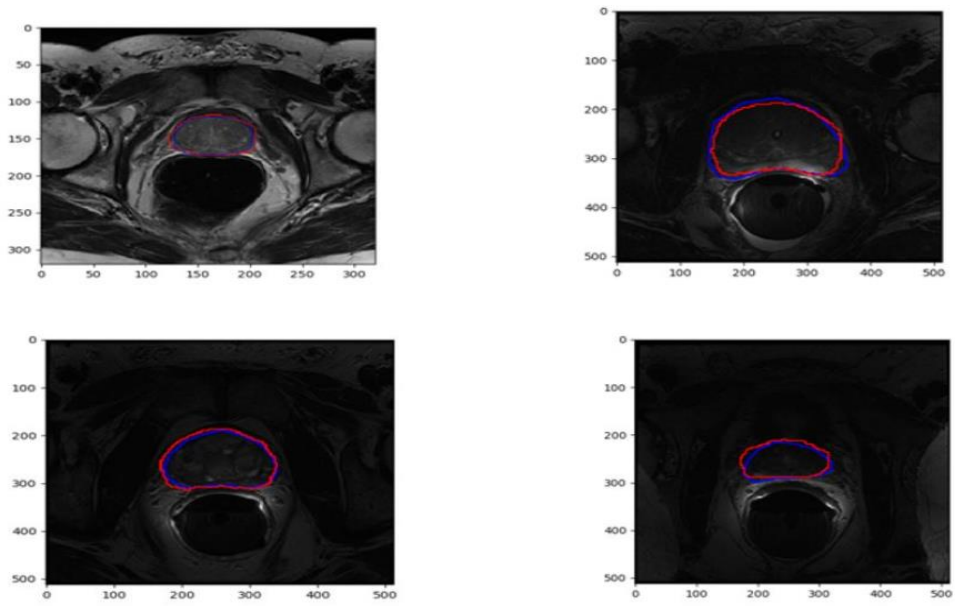

Figure 12: Prostate manually segmented, coloured in (blue) and automatically segmented, colored in (red)

3.2. Quantitative evaluation results

To evaluate the quantitative performance of the proposed segmentation algorithm, the calculated Dice Similarity Coefficient gives a maximum of 89.04% and a minimum of 70.75% and a mean DSC of 84.92% as shown in Table 3; while the HD gives a maximum of 28.4, the minimum of 1.96 and the mean Hausdorff of 5.3mm as plotted in the figure 13.

Table 3: The quantitative results, DSC values

Fold	1	2	3	4	5	6	7	8	9	10	**Mean**
DSC (%)	83.64	70.75	87.12	88.3	86.54	87.96	83.85	87.29	84.79	89.04	**84.92**

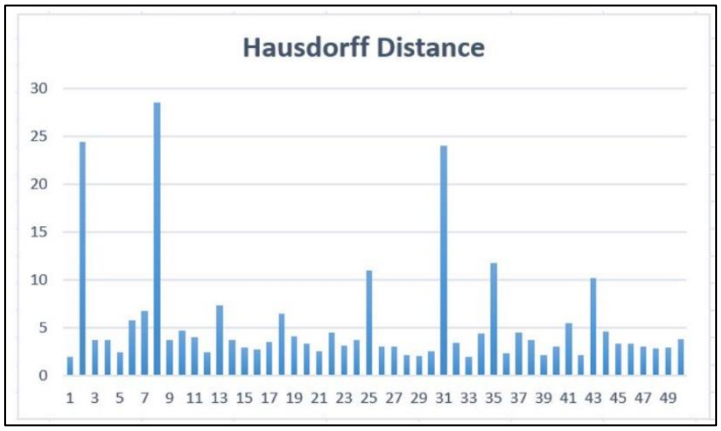

Figure 13: Graphical representation of Hausdorff Distance

https://dx.doi.org/10.4314/rjeste.v6i1.4

3.3. Comparison of the proposed methods with other methods

As we have shown in Table 4, the proposed algorithm was evaluated with a maximum Dice score of 89.04% and a mean Dice score of 84.92%. (Maan et al., 2012) proposed an automatic prostate segmentation algorithm from transversal T2 weighted images based on 3D Active Appearance Models (AAM). The model performance was evaluated to the mean Dice Score of 78%. (Zhang et al., 2017) introduced an automatic pipeline to segment the prostate in diffusion MRI, separating the prostate gland from the surrounding organs. The separation of the prostate gland was followed by a post-processing phase through active contours. This model proposed by Zhang was trained and evaluated on 25 clinical images of patients and achieved an overall accuracy with a mean Dice score of 84%. The study done by Rundo et al. (Rundo et al., 2019) proposed a new whole prostate gland segmentation algorithm based on U-Net Architecture by introducing squeeze-and-excitation (SE) blocks in the decoder and encoder of the U-Net architecture. During the study conducted by (Khan et al., 2020) to evaluate four encoder-decoder CNNs in the segmentation of the prostate gland in T2-Weighted MR images, he used the patch-wise DeepLabV+3 where the researcher achieved an accuracy with Dice score of 78.9% to segment the whole prostate gland. See Table 4 summarizes the comparison of our proposed segmentation algorithm with other published algorithms.

Table 4: Comparison of the proposed model with other published models

The model proposed by:	Dice score (%)
Khan et al.[12]	78.9 (%)
Rundo et al.[24]	76 (%)
Maan et al. [18]	78 (%)
Zhang et al.[30]	84 (%)
Our proposed model	**84.9 (%)**

4. Results Discussion

By performing an automatic prostate segmentation task on a relatively small data set using deep learning tools, we built an automatic deep learning model that achieves promising results for MRI. In the same regard, the model was evaluated on the Promise 12 data set and showed outperforming results compared to other segmentation methods mentioned in our literature.
The table 3 shows the Dice Score for each of the 10 folds of the data set with the overall Mean Dice score averaged over all the presented 10 folds. As it is commonly accepted that a Dice score value greater than 0.7 (70%) represents a good agreement, the Mean Dsc for our segmentation whose value is 84.92% proves that the proposed model achieved good performance with promising results.

https://dx.doi.org/10.4314/rjeste.v6i1.4

Note that, although our Mean DCS value (i.e 84.92%) is greater than values reported in the research conducted by Khan et al., 2020) (Mean Dsc: 78.9%), (Rundo et al., 2019) (Mean Dsc: 76%), (Maan et al., 2012) (Mean Dsc: 78%) and (Zhang et al., 2017) (Mean Dsc:84%); the proposed model is also lower compared to some other models, especially models built upon the combination of different methods, for instance, (Cheng et al., 2018) proposed an enhanced HNN model for the automatic segmentation of prostate in MRI, which has achieved good results with a mean Dsc of 89.77%.

5. Conclusions

Prostate cancer is the most common type of cancer often diagnosed in men all over the world. However, when diagnosed at its early stage, this PCa can be cured. The main goal of this research study was to propose a deep learning-based automatic segmentation method of prostate from MRI, as a new tool that can assist radiologists and medical doctors for prostate cancer diagnosis and treatment decisions.

In this work, the proposed method was developed by using U-Net architecture and was trained and evaluated on a promise12 data set that consists of 50 T2-Weighted three-dimensional magnetic resonance images (3D MRIs) made publically available by MICCAI Grand Challenge team for prostate segmentation purposes. The 10-fold cross-validation approach was used to train and evaluate the proposed method. The Dice Score Coefficient (DSC) and Hausdorff Distance (HD) were used as quantitative evaluation metrics. The results showed the lowest DSC of 70.75 and the highest DSC of 89.04, then a mean DSC of 84.92. The computed HD is 5.3mm. The proposed method was also evaluated by comparing it with other segmentation methods in the literature where it was shown to be outperforming them.

In conclusion, the proposed prostate segmentation method achieved good accuracy with promising results even better than other segmentation methods due to using U-Net model that can achieve the best accuracy on medical image segmentation with relatively small data sets.
The purpose of the proposed method was to perform automatic segmentation of the prostate from MRI. Along with the research study, we have noticed that various directions can help for the future improvement of this study. Performing the same study on a large data set and combining U-Net with other segmentation methods can make the used method more robust and achieve higher accuracy.

For future work, we recommend creating an interface for this proposed method, whereby the users will be able to load images and perform the segmentation on the interface. This will help to save time taken by manual segmentation reduce errors observed during manual segmentation and will help the country to save money that is paid to external experts who are paid to perform manual segmentation in the old system.

6. References

Adeloye, David, Aderemi, Iseolorunkanmi, Oye-dokun, Iweala, Omoregbe, & K Ayo. (2016). An estimate of the incidence of prostate cancer in Africa: a systematic review and meta-analysis.

https://dx.doi.org/10.4314/rjeste.v6i1.4

Bong, Liew, & Lam (2016). Ground-glass opacity nodules detection and segmentation using the snake model. In *Bio-Inspired Computation and Applications in Image Processing*, pages 87–104. Elsevier.

Chen, Ball, Yang, Moradzadeh, Chapman, Larson, Langlotz, Amrhein, & Lungren. (2018). Deep learning to classify radiology free-text reports.

Christ, Elshaer, Ettlinger, Tatavarty, Bickel, Bilic, Rempfler, Armbruster, Hofmann, D'Anastasi. (2016). Automatic liver and lesion segmentation in ct using cascaded fully convolutional neural networks and 3d conditional random fields.

Esteva, Kuprel, Novoa, Ko, Swetter, Blau, & Thrun. (2017). Dermatologist-level classification of skin cancer with deep neural networks.

Gillespie, Kendrick, Boon, Boon, Rattay, & Yap. (2020). Deep learning in magnetic resonance prostate segmentation: A review and a new perspective.

Gulshan, Peng, Coram, Stumpe,Wu, Narayanaswamy, Venugopalan, Widner, Madams, & Cuadroset. (2016). Development and validation of a deep learning algorithm for the detection of diabetic retinopathy in retinal fundus photographs.

Guo, Gao, & Shen. (2013). Deformable Mr Prostate segmentation via deep feature learning and sparse patch matching.

Habes, Schiller, Rosenberg, Burchardt, &W. (2013). Hoffmann. Automated prostate segmentation in whole-body MRI scans for epidemiological studies. .

Huttenlocher, Klanderman, & Rucklidge. (1993). Comparing images using the Hausdorff distance.

Jadon. (2020). A survey of loss functions for semantic segmentation.

Khan,Yahya, Alsaih, Ali, & Meriaudeau. (2020). Evaluation of deep neural networks for semantic segmentation of the prostate in t2w mri.

Kim, Choi, & Park. (2018). Improving arterial spin labelling by using deep learning.

Krizhevsky, Sutskever, & Hinton. (2012). Imagenet classification with deep convolutional neural networks.

Lakhani, & Sundaram. (2017). Deep learning at chest radiography: automated classification of pulmonary tuberculosis by using convolutional neural networks.

Liao, Gao, Oto, & Shen. (2013). Representation learning: A unified deep learning framework for automatic prostate mr segmentation.

Liu, Jang, Kijowski, Bradshaw, & McMillan. (2018). Deep learning mr imaging–based attenuation correction for pet/mr imaging.

Maan & Heijden. (2012). Prostate mr image segmentation using 3d active appearance models.

Maas, Hannun, & Ng. (2013). Rectifier nonlinearities improve neural network acoustic models.

Milletari, Navab, & Ahmadi. (2016). V-net: Fully convolutional neural networks for volumetric medical image segmentation.

Plewes & Kucharczyk. (2012). Physics of mri.

Razzak, Naz, & Zaib. (2018). Deep learning for medical image processing: Overview, challenges and the future.

Ronneberger, Fischer, & Brox. (2015). U-net: Convolutional networks for biomedical image segmentation.

https://dx.doi.org/10.4314/rjeste.v6i1.4

Rundo, Han, Nagano, Zhang, Hataya, Militello, Tangherloni, Nobile, Ferretti, & Besozzi. (2019). Use-net: Incorporating squeeze-and-excitation blocks into u-net for prostate zonal segmentation of multi-institutional mri dataset.

Russakovsky, Deng, Su, Krause, Satheesh, Ma, Huang, Karpathy, Khosla, & Bernstein. (2015). Imagenet large scale visual recognition challenge.

Suzuki. (2017). Survey of deep learning applications to medical image analysis.

Tian, Li, Chen, Zheng, Fan, Li, Li, & Du. (2021). Interactive prostate mr image segmentation based on convlstms and ggnn.

Yamashita, Nishio, Do, & Togashi. (2018). Convolutional neural networks: an overview and application in radiology.

Yasaka, Akai, Abe, & Kiryu. (2018). Deep learning with convolutional neural network for differentiation of liver masses at dynamic contrast-enhanced ct: a preliminary study.

Zhang, Baig, Wong, Haider, & Khalvati. (2017). Segmentation of prostate in diffusion mr images via clustering.

https://dx.doi.org/10.4314/rjeste.v6i1.4